



**Centro Universitário Farias Brito**  
**Bacharelado em Ciência da Computação**  
**TCC**

**Simulador de Algoritmos de Criptografia de  
Cifra de Blocos com Finalidade Educacional**

**Tanielian Viana Barreira**

**Me. Sérgio Araújo Yunes**

Tanielian Viana Barreira

# **Simulador de Algoritmos de Criptografia de Cifra de Blocos com Finalidade Educacional**

Trabalho de Conclusão de Curso para o curso de Ciência da Computação do Centro Universitário Farias Brito.

Centro Universitário Farias Brito  
Bacharelado em Ciência da Computação  
TCC

Orientador(a): Me. Sérgio Araújo Yunes

Fortaleza-CE, Brasil

2020

*Dedico esta labuta à todas as pessoas que sempre me zoaram...*

# Agradecimentos

Agradeço ao meu pai...

Por último mas não menos importantes, muito pelo contrário, aos meus professores...

# Resumo

TODO: Resumo

**Palavras-chave:** criptografia, educacional

# Abstract

TODO: Abstract

**Keywords:** cryptography, educational



# Lista de Siglas e Abreviações

**3DES** *Triple Data Encryption Standard.* 13

**AES** *Advanced Encryption Standard.* 13

**CCSDS** *Consultive Committee for Space data Systems.* 14

**DEA** *Data Encryption Algorithm.* 13

**DES** *Data Encryption Standard.* 13, 15, 23–25

**IAB** *Internet Architecture Board.* 16

**IBM** *International Business Machines Corporation.* 13

**S-DES** *Simplified Data Encryption Standard.* 8, 25–28, 31, 35, 47

**TCC** *Trabalho de Conclusão de Curso.* 47



# Lista de ilustrações

Figura 1 – Criptografia Simétrica . . . . .	18
Figura 2 – Exemplo de <i>Rail fence</i> . . . . .	18
Figura 3 – Exemplo de cifra de César . . . . .	18
Figura 4 – Imagem da máquina Enigma . . . . .	19
Figura 5 – Fluxo da mensagem criptografada quando o foco é confidencialidade . .	20
Figura 6 – Fluxo da mensagem criptografada quando o foco é autenticidade . . .	21
Figura 7 – Fluxo da mensagem criptografada quando o foco é confidencialidade e autenticidade . . . . .	21
Figura 8 – Cifra de bloco ideal . . . . .	22
Figura 9 – S-DES <i>scheme</i> . . . . .	26
Figura 10 – Geração das Chaves do S-DES . . . . .	27
Figura 11 – Detalhamento da criptografia do S-DES . . . . .	28
Figura 12 – Estrutura da interface . . . . .	32
Figura 13 – Cabeçalho . . . . .	32
Figura 14 – Conteúdo . . . . .	33
Figura 15 – Rodapé . . . . .	33
Figura 16 – Visualização em um <i>browser</i> de dispositivo móvel . . . . .	34
Figura 17 – Tela inicial - entrada de dados . . . . .	35
Figura 18 – Etapa P10 - Permutação de 10 bits . . . . .	36
Figura 19 – Etapa LS-1 - <i>Circular Left Shift</i> de 1 posição . . . . .	37
Figura 20 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-1 . . . . .	37
Figura 21 – Chave $K_1$ - Geração da primeira chave $K_1$ . . . . .	38
Figura 22 – Etapa LS-2 - <i>Circular Left Shift</i> de 2 posições . . . . .	38
Figura 23 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-2 . . . . .	39
Figura 24 – Chave $K_2$ - Geração da segunda chave $K_2$ . . . . .	39
Figura 25 – Passo IP ( <i>Initial Permutation</i> ) - Permutação Inicial . . . . .	40
Figura 26 – L (esquerda) & R (direita) . . . . .	40
Figura 27 – Passo $f_K$ - Função que usa uma chave . . . . .	40
Figura 28 – Etapa E/P - Permutação de Expansão . . . . .	41
Figura 29 – Etapa XOR - OU exclusivo com $K_1$ . . . . .	42
Figura 30 – Etapa S0 & S1 - Substituições S0 e S1 . . . . .	43
Figura 31 – Etapa P4 - Permutação de 4 bits . . . . .	44
Figura 32 – Etapa XOR - OU exclusivo com L . . . . .	44
Figura 33 – Resultado de $f_K$ . . . . .	45
Figura 34 – Passo SW - Troca . . . . .	45
Figura 35 – Passo $IP^{-1}$ - Permutação Inicial Inversa . . . . .	46

Figura 36 – Resultado . . . . .	47
Figura 37 – <i>Feedback request</i> . . . . .	47
Figura 38 – Início do questionário . . . . .	49
Figura 39 – Enquadramento do usuário . . . . .	50
Figura 40 – Conhecimento antes da utilização do simulador . . . . .	50
Figura 41 – Conhecimento depois da utilização do simulador . . . . .	51
Figura 42 – Efetividade do aprendizado . . . . .	51
Figura 43 – Melhorias nas descrições das etapas . . . . .	52
Figura 44 – Melhorias nas execuções das etapas . . . . .	53

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>12</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>15</b>
<b>2.1</b>	<b>História</b>	<b>15</b>
<b>2.2</b>	<b>Criptografia</b>	<b>16</b>
2.2.1	Criptografia Simétrica	17
2.2.2	Criptografia Assimétrica	19
<b>2.3</b>	<b>Cifra de blocos</b>	<b>22</b>
<b>2.4</b>	<b>DES</b>	<b>23</b>
<b>2.5</b>	<b>3DES</b>	<b>24</b>
<b>2.6</b>	<b>S-DES</b>	<b>25</b>
<b>2.7</b>	<b>Ferramentas de apoio ao ensino</b>	<b>28</b>
<b>2.8</b>	<b>Simuladores voltados ao ensino</b>	<b>29</b>
<b>3</b>	<b>FERRAMENTA DESENVOLVIDA</b>	<b>31</b>
<b>3.1</b>	<b>Informações técnicas</b>	<b>31</b>
<b>3.2</b>	<b>Usuário</b>	<b>31</b>
<b>3.3</b>	<b>Estrutura da interface</b>	<b>32</b>
3.3.1	Cabeçalho	32
3.3.2	Conteúdo	33
3.3.3	Rodapé	33
3.3.4	<i>Mobile</i>	34
<b>3.4</b>	<b>Interface do conteúdo</b>	<b>34</b>
3.4.1	Tela inicial - entrada de dados	35
3.4.2	Passo Chaves - Geração das chaves	36
3.4.2.1	Etapa P10 - Permutação de 10 bits	36
3.4.2.2	Etapa LS-1 - <i>Circular Left Shift</i> de 1 posição	37
3.4.2.3	Etapa P8 - Permutação de 8 bits sobre o resultado de LS-1	37
3.4.2.4	Resultado Chave $K_1$	38
3.4.2.5	Etapa LS-2 - <i>Circular Left Shift</i> de 2 posições	38
3.4.2.6	Etapa P8 - Permutação de 8 bits sobre o resultado de LS-2	39
3.4.2.7	Resultado Chave $K_2$	39
3.4.3	Passo IP - Permutação Inicial	40
3.4.4	Passo $f_K$ - Função que usa uma chave	40
3.4.4.1	Etapa E/P - Permutação de Expansão	41
3.4.4.2	Etapa XOR - OU exclusivo com $K_1$	42

3.4.4.3	Etapa S0 & S1 - Substituições S0 e S1 . . . . .	43
3.4.4.4	Etapa P4 - Permutação de 4 bits . . . . .	44
3.4.4.5	Etapa XOR - OU exclusivo com L . . . . .	44
3.4.4.6	Resultado de $f_K$ . . . . .	45
3.4.5	Passo SW - Troca . . . . .	45
3.4.6	Passo $f_K$ - Função que usa a outra chave . . . . .	46
3.4.7	Passo $IP^{-1}$ - Permutação Inicial Inversa . . . . .	46
3.4.8	Tela final - Resultado . . . . .	47
<b>3.5</b>	<b>Limites da solução . . . . .</b>	<b>47</b>
<b>3.6</b>	<b>Comparação com outros simuladores criptográficos . . . . .</b>	<b>48</b>
<b>4</b>	<b>PESQUISA COM OS ALUNOS . . . . .</b>	<b>49</b>
<b>4.1</b>	<b>Perguntas . . . . .</b>	<b>49</b>
4.1.1	Enquadramento do usuário . . . . .	50
4.1.2	Conhecimento antes e depois da utilização do simulador . . . . .	50
4.1.3	Efetividade do aprendizado . . . . .	51
4.1.4	Melhorias no simulador . . . . .	51
<b>4.2</b>	<b>Respostas . . . . .</b>	<b>53</b>
4.2.1	Enquadramento do usuário . . . . .	53
4.2.2	Conhecimento antes e depois da utilização do simulador . . . . .	53
4.2.3	Efetividade do aprendizado . . . . .	53
4.2.4	Melhorias no simulador . . . . .	54
<b>5</b>	<b>CONCLUSÕES E TRABALHOS FUTUROS . . . . .</b>	<b>55</b>
<b>5.1</b>	<b>Extensão do Simulador . . . . .</b>	<b>55</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>56</b>

# 1 Introdução

A formação de pessoas na área de Ciência da Computação exige que as ferramentas se aproximem ao máximo da realidade existente. A academia apresenta modelos da realidade que são possíveis de manuseio no ambiente de estudo. A visibilidade que o aprendiz precisa ter sobre o assunto em estudo deve ser semelhante àquela que será encontrada durante o seu exercício profissional. E quando essa similaridade não é encontrada, se torna difícil para o aluno visualizar corretamente em que cenário se aplica o conhecimento adquirido quando este precisar ser utilizado (MAIA, 2001) (MAIA; PACHECO, 2003) (SILVA, 2009).

As atuais necessidades de aplicação da Ciência da Computação exigem daqueles que desenvolvem sistemas, conhecimento detalhado de como funcionam as ferramentas de trabalho, em especial os métodos e técnicas da Criptografia. Essa necessidade decorre do fato de que a *Internet*, definitivamente, está inserida na vida cotidiana das pessoas e novos costumes estão sendo adotados pela sociedade. O comércio eletrônico, as operações bancárias de transferência eletrônica de fundos, o uso do cartão de crédito como moeda de plástico, por exemplo, se constituem na realidade do convívio social. Junte-se a isso o fato de que a mobilidade na telefonia é uma realidade cristalizada entre as pessoas, independente de qual seja a sua posição social.

Por outro lado, devido à alta complexidade dos algoritmos usados nas aplicações da Criptografia (SILVA, 2009), a transmissão do conhecimento nem sempre é eficiente o suficiente, principalmente se os algoritmos forem ensinados na sua complexidade real.

O ensino nas disciplinas que envolvam a temática da segurança da informação, abordando mais especificamente o conteúdo da Criptografia – preocupação prioritária desta pesquisa – acaba sendo prejudicado pela falta de tempo hábil para discorrer sobre o assunto e de profissionais capacitados. (TODO: REFERÊNCIA) Naturalmente, a abordagem de cunho pedagógico precisa preservar os aspectos fundamentais da arquitetura de cada algoritmo.

Embora se saiba que, em nome da preservação da eficiência do trabalho acadêmico os algoritmos possam ser apresentados com alguma restrição na sua abrangência, ainda assim, é possível apresentar todo o funcionamento e implementação dos algoritmos durante o período do desenvolvimento de uma disciplina, quer no tempo destinado a uma disciplina de graduação, quer de pós-graduação, com um esforço menor do que aquele necessário para a implementação do algoritmo em sua forma original (MAIA, 2001) (MAIA; PACHECO, 2003).

Analisando o contexto acima apresentado, deriva-se a necessidade de um instrumento que auxilie o processo de ensino nesta área. A finalidade deste projeto é o

desenvolvimento e a implementação uma ferramenta que possibilitará a simulação do funcionamento da versão simplificada do algoritmo criptográfico de chave simétrica conhecido como *Data Encryption Standard* (DES), para utilização de natureza pedagógica.

O produto derivado do desenvolvimento será um simulador destinado a apoiar a formação de profissionais que irão se utilizar de aplicações que utilizam algoritmos de criptografia por chave simétrica.

Esse simulador permitirá, então, que o algoritmo possa ser executado de modo ininterrupto, ou por etapas discretas, de modo a permitir a completa compreensão do seu funcionamento.

Diante do contexto apresentado, ressalta-se que o propósito desta monografia é contribuir para a resolução da problemática posta, motivo pelo qual define-se como objetivo geral o desenvolvimento de uma ferramenta de simulação de algoritmo criptográfico com intuito de auxiliar o processo de ensino-aprendizagem da técnica de cifra de blocos, nas disciplinas que abordem a criptografia.

Em decorrência, os objetivos específicos tem o compromisso de investigar, através de um questionário, quais as etapas dos algoritmos criptográficos baseados em cifra de blocos que geram mais dificuldades no aprendizado dos alunos; definir requisitos funcionais da ferramenta; desenvolver a ferramenta de simulação para os algoritmos *Data Encryption Standard* (DES) e *Triple Data Encryption Standard* (3DES).

Considerando que o desenvolvimento de uma ferramenta de simulação de algoritmos criptográficos é o desafio maior da pesquisa apresentada, é importante ressaltar que historicamente a criptografia por chave secreta, (utilizada nessa ferramenta), experimentou um grande impulso por volta do ano de 1974, quando foi apresentado o algoritmo *Data Encryption Standard* (DES). Trata-se de um método para a criptografia de dados baseado em cifra de blocos, que se tornou o padrão usado pelo público e, em particular, pelo governo dos Estados Unidos. Alguns documentos fazem uma distinção entre o DES como um padrão, se referindo ao algoritmo de sua implementação como *Data Encryption Algorithm* (DEA).

Esse algoritmo herdou os princípios da Cifra de *Feistel*, (1973) resultado de um projeto desenvolvido pela *International Business Machines Corporation* (IBM) sobre Criptografia por cifra de blocos. Apesar dos questionamentos sobre a sua vulnerabilidade, por conta do tamanho da chave, o entendimento de como funciona o DES transmite importante conhecimento sobre o mecanismo utilizado nas cifras de bloco.

No ano de 2001, depois de um trabalho de cinco anos, o *Advanced Encryption Standard* (AES), conhecido pela sua implementação mais famosa '*Rijndael*', em alusão aos seus criadores, os belgas *Joan Daemen* e *Vincent Rijmen*, passou a ser o novo padrão utilizado para a Criptografia de dados. Trata-se de um método criptográfico também baseado em cifra de bloco e que, tal qual o seu antecessor DES, espera-se da comunidade

científica que seja utilizado e detalhadamente analisado.

Dentre as principais características do *Rijndael* se encontra o fato de que o algoritmo ocupa pouca memória, o que o qualifica para ser utilizado em ambientes restritos, tais como telefones celulares e *smart cards*. Justamente por essas características de restrição de necessidades, esse algoritmo também é recomendado pelo *Consultive Committee for Space data Systems* (CCSDS), organização da qual faz parte o proponente do projeto.

As áreas Empresarial e Acadêmica, que se utilizam de profissionais criados na academia, possuem necessidades criptográficas (segurança nas empresas e didática nas academias) não supridas atualmente devido a falta de uma metodologia eficiente de transmissão do conhecimento para o aluno e pela dificuldade inerente a área de conhecimento citada (SILVA, 2009).

TODO: Parágrafo de finalização... Precisa?

## 2 Fundamentação Teórica

Inicia-se com o tópico 5.1 História, onde há uma breve contextualização histórica sobre segurança e criptografia. No tópico 5.2 Criptografia, é feita uma visão geral sobre criptografia e suas aplicabilidades.

Os dois seguintes 5.2.1 Criptografia Simétrica e 5.2.2 Criptografia Assimétrica tratam de dois tipos de criptografia e explanam sobre suas definições, seus cenários de utilização, e trazem alguns exemplos de algoritmos sobre seus respectivos ramos da criptografia. O tópico 5.3 Cifra de Blocos fala sobre esse método de criptografia simétrica no qual é feita uma visão geral sobre seu funcionamento e benefícios obtidos por sua utilização.

Os tópicos 2.4 DES e 2.5 3DES tratam sobre estes algoritmos de criptografia simétrica baseados em cifra de blocos, os quais trazem uma visão geral e uma breve explicação de cada algoritmo.

No tópico 5.6 Ferramentas de apoio ao ensino, é dada uma visão geral sobre o que são e sobre o ganho que elas trazem para os alunos. O último tópico, 5.7 Simuladores voltados ao ensino, traz uma visão geral sobre simuladores e uma análise da existência de simuladores voltados ao ensino, tanto de uma forma geral como de criptografia.

Neste caso, é perceptível a intenção de trazer uma clara e breve explicação sobre os tópicos abordados neste trabalho, de forma que este se faça compreender da melhor forma possível.

### 2.1 História

Há algumas décadas atrás, antes que se fosse comum o uso de equipamentos de processamento de dados, a segurança da informação que era considerada importante para uma empresa se dava basicamente através de dois meios: o administrativo e o físico. Um bom exemplo para o meio administrativo é o uso de um processo de aquisição de profissionais bastante seletivo e rigoroso, assim como a utilização de um contrato de confidencialidade que protege a empresa de possíveis ‘vazamentos’ de dados. Um bom exemplo para o meio físico é o uso de cofres e senhas para armazenar documentos importantes ou até confidenciais.

Com o surgimento do computador surgiu a necessidade de proteger virtualmente os arquivos, agora armazenados de forma virtual no computador. Essa necessidade é ainda mais evidente com o surgimento da Internet, que traz uma facilidade de comunicação muito grande entre os computadores.



O *Internet Architecture Board* (IAB) emitiu, em 1994, um relatório de título “Security in the internet architecture” (Segurança na arquitetura da Internet). O documento estabelecia que a internet necessitava de mais e melhor segurança. Entre as principais áreas citadas no relatório como sendo as que mais necessitavam de segurança estavam a infraestrutura da rede contra monitoração e controle não autorizados do tráfego da rede e também a necessidade de proteger o tráfego entre usuários finais se utilizando de autenticações e criptografias.

Com o passar dos anos, os ataques através da Internet se tornaram mais evoluídos, eles se tornaram mais automatizados e mais devastadores, necessitando cada vez mais de formas de segurança também mais evoluídas (STALLINGS, 2014).

Existe uma lista de mecanismos de segurança explicitados na recomendação X.800 (ITU, 1991). Entre eles temos a cifragem, que é melhor descrita mais adiante. A X.800 divide, muito claramente, dois tipos de criptografia, a reversível e a irreversível. A criptografia reversível é composta por um algoritmo matemático que permite que dados sejam criptografados e que estes dados criptografados possam ser decriptografados posteriormente. Já a criptografia irreversível, por sua vez, é capaz de criptografar os dados, mas a decriptografia desses dados é impossível. Esta por sua vez tem objetivos diferentes da reversível, que visa somente trafegar ou armazenar dados de maneira segura, ela é composta de algoritmos de hash e tem como objetivo autenticação e assinatura digital (STALLINGS, 2014) (ITU, 1991).

## 2.2 Criptografia

Segundo a National Research Council (1991, apud STALLINGS, 2008, pg. 15), “A criptografia provavelmente é o aspecto mais importante da segurança de comunicações e está se tornando cada vez mais importante como um componente básico para a segurança do computador.”.

O termo Criptografia vem do Grego *kryptós*, que significa “escondido” e de *gráphein*, que significa “escrita”. Ela é o estudo dos princípios e técnicas pelas quais os dados podem ser transformados da sua forma original em outra ilegível. Dessa forma, os dados podem ser conhecidos somente pelo destinatário, o detentor da “chave secreta”, o que faz com que mesmo que tenham sido interceptados, torne difícil a leitura de seu conteúdo por alguém não autorizado. Ela faz parte da Criptologia e é um sub-ramo da Matemática (KNUDSEN, 1998).

O estudo da maneira de camuflar o real significado de uma mensagem usando técnicas e algoritmos de cifragem têm evoluído juntamente com o estudo da maneira de se conseguir entender a mensagem quando não se é o real destinatário da mesma. Este campo de estudo é chamado Criptoanálise (GAINES, 1956). A Criptologia engloba a Criptografia

e a Criptoanálise. Alguns autores se utilizam do termo Criptovirologia quando falam de vírus que se utilizam de chaves publicas (YOUNG; YUNG, 2004).

Há também a Esteganografia que não faz parte de Criptologia, mesmo sendo estudada em situações bem similares e até pelos mesmos autores. Ao contrário da criptografia que modifica a informação com intuito de transformar seu estado original em algo indecifrável a Esteganografia estuda formas de como se pode camuflar uma informação dentro de outra. Temos também a Esteganálise que está para Esteganografia assim como a Criptoanálise está para a Criptografia (SALOMON, 2005).

A criptografia se divide em Simétrica, também chamada de criptografia convencional, e Assimétrica, também chamada de criptografia por chave pública (STALLINGS, 2014) e dentro dessa divisão ainda temos algoritmos de criptografia reversíveis e irreversíveis (STALLINGS, 2014) (ITU, 1991).

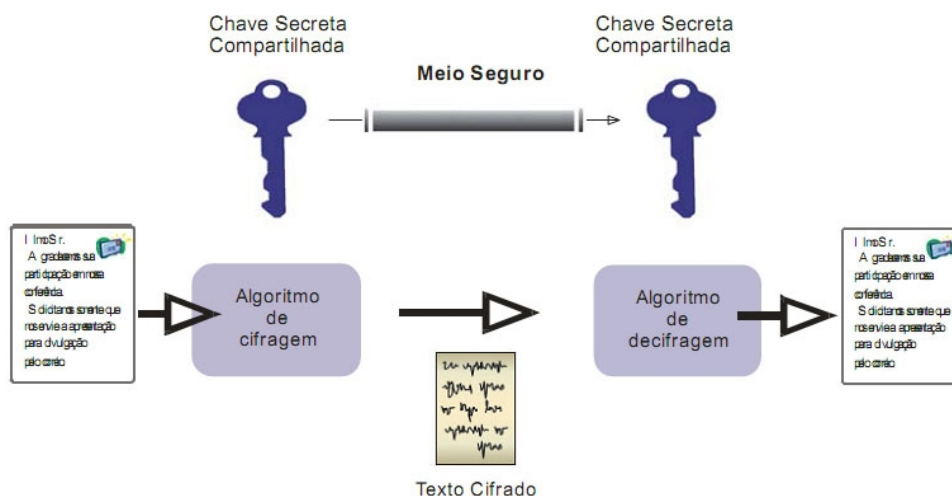
A criptografia possui inúmeras áreas de aplicação. Desde transações bancárias, envio de e-mails, segurança de arquivos sigilosos, segurança de autenticação, armazenamento de senhas em bancos de dados até o sinal de linha telefônica, sinal de TV digital e acesso a sites certificados (AVELINO; AVELINO, 2007).

### 2.2.1 Criptografia Simétrica

Também chamada de criptografia convencional, a criptografia simétrica é um criptosistema onde tanto a criptografia como a decriptografia são realizadas com a mesma chave.

Ela transforma uma mensagem clara em uma mensagem cifrada, usando um algoritmo de criptografia e uma chave. Esta mesma chave juntamente com a inversão do algoritmo utilizado (gerando assim o algoritmo de decriptografia do algoritmo), são necessários para que se possa obter a mensagem clara novamente a partir da mensagem cifrada, assim como demonstra a Figura 1.

Figura 1 – Criptografia Simétrica



Fonte: "Introdução à certificação digital: da criptografia ao carimbo de tempo"(BROCARD; ROLT; FERNANDES, 2006)

Antes do computador, as cifras simétricas tradicionais poderiam se utilizar de técnicas de transposição ou de substituição, ou até as duas técnicas combinadas. Essas técnicas são os componentes básicos para todas as técnicas de criptografia.

Técnicas de transposição transpõem sistematicamente as posições dos elementos da mensagem clara. Tal técnica consiste na aplicação de alguma permutação na mensagem clara de forma que a mensagem final seja ilegível, e somente o detentor da forma de como os elementos da mensagem foram permutados pode obter novamente a mensagem de forma clara. A cifra mais simples dessa técnica é a cifra de *Rail fence*. A Figura 2 ilustra como ficaria a crifa *Rail fence* de profundidade dois do texto “Técnica de transposição”.

Figura 2 – Exemplo de *Rail fence*

T c i a e r n p s ç o  
é n c d t a s o i ã

Técnicas de substituição mapeiam elementos, caracteres ou bits, da mensagem clara e as substitui por outras letras ou números ou até símbolos. Analisando a mensagem clara como sendo um conjunto de bits, então a substituição é definida pela troca de padrões de bits da mensagem clara por padrões de bits da mensagem cifrada. Como exemplo clássico temos a cifra de César, criada por Júlio César. A Figura 3 ilustra como ficaria a cifra de César com o texto “Tecnica de substituiçao”.

Figura 3 – Exemplo de cifra de César

Alfabeto:  
Claro: a b c d e f g h i j k l m n o p q r s t u v w x y z  
Cifra: t u v w x y z a b c d e f g h i j k l m n o p q r s  
Mensagem clara: Tecnica de substituiçao  
Mensagem cifrada: Mxvgbvt wx lnu lmbmnbvth

Existem basicamente dois os ataques possíveis em um algoritmo criptográfico. A criptoanálise, que se baseia nas características do próprio algoritmo de criptografia, e a força bruta, que engloba simplesmente repetidas tentativas de todas as chaves possíveis até encontrar a correta.

Antes da existência do computador existiam máquinas que implementavam a nível de hardware técnicas de substituição. Eram conhecidas como máquinas de rotor. Duas foram as máquinas de rotor mais conhecidas, a da Alemanha, conhecida como Enigma e a do Japão conhecida como Purple. Elas foram utilizadas durante a segunda guerra mundial e a quebra desses dois códigos pelos Aliados foi significativa para o resultado da guerra. Abaixo, a Figura 4 mostra a máquina Enigma com três rotores.

Figura 4 – Imagem da máquina Enigma



Máquina Enigma com três rotores, teclado, luzes e conexões para câmbio de codificação. Fonte: Wikipedia (WIKIPÉDIA, 2020)

Algoritmos simétricos são utilizados em cenários onde a mensagem tem uma necessidade de ser decryptografada, por exemplo, o Kerberos, serviço de autenticação desenvolvido no *Massachusetts Institute of Technology* (MIT) como parte do projeto Athena, que visa trazer segurança a cenários distribuídos abertos, se utiliza unicamente de cifra simétrica (STALLINGS, 2014).

### 2.2.2 Criptografia Assimétrica

Também chamada de criptografia por chave pública, a criptografia assimétrica é um criptosistema onde a criptografia e a decryptografia são realizadas com chaves diferentes, uma pública e outra privada. Quando uma mensagem é criptografada com uma chave,

somente a outra chave poderá decryptografar a mensagem. O criptosistema assimétrico mais utilizado é o RSA.

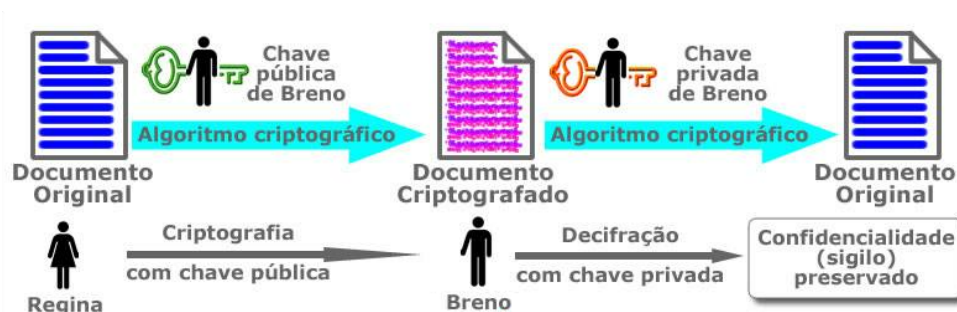
Ela transforma uma mensagem clara em uma mensagem cifrada utilizando uma das duas chaves acima citadas e um algoritmo de criptografia. E utilizando a outra chave citada acima e o algoritmo de decryptografia, é possível extrair a mensagem clara a partir da mensagem cifrada.

A criptografia assimétrica possui três utilizações básicas: confidencialidade, autenticação ou ambas.

A confidencialidade garante que somente o destinatário será capaz de ler a mensagem. Quando se tem essa necessidade, deve-se criptografar a mensagem com a chave pública do destinatário, enviar a mensagem criptografada para o destinatário, e ele, de posse da chave privada, poderá decryptografar a mensagem e dessa forma obter a mensagem original.

O fato do destinatário ter conseguido decryptografar a mensagem com a chave privada dele próprio garante que a mensagem só poderia ter sido decryptografada por ele, detentor da chave privada, mas não garante a identidade do remetente, visto que a chave utilizada para criptografar a mensagem é pública, assim como demonstra a Figura 5.

Figura 5 – Fluxo da mensagem criptografada quando o foco é confidencialidade



Fonte:

A autenticidade garante que quem enviou a mensagem foi o remetente. Quando se tem essa necessidade, deve-se criptografar a mensagem com a chave privada do remetente, enviar a mensagem criptografada para o destinatário, e ele, de posse da chave pública do remetente, poderá decryptografar a mensagem e assim obter a mensagem original.

O fato do destinatário ter conseguido decryptografar a mensagem com a chave pública do remetente comprova que a mensagem é realmente do remetente, mas não garante que o destinatário é o único que poderia decryptografar a mensagem, visto que qualquer pessoa pode possuir a chave pública do remetente, assim como demonstra a Figura 6.

Figura 6 – Fluxo da mensagem criptografada quando o foco é autenticidade

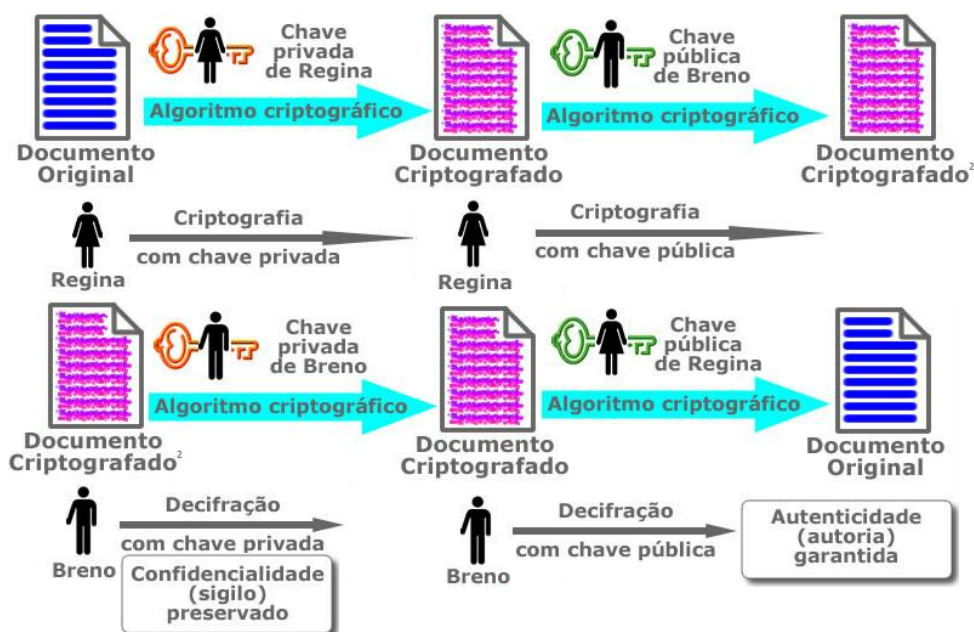


Fonte:

É possível também que se criptografe a mensagem com a chave privada do remetente, criptografar a mensagem já criptografada com a chave pública do destinatário, enviar a mensagem duplamente criptografada para o destinatário, e ele de posse da chave privada, poderá decryptografar a mensagem duplamente criptografada em uma mensagem criptografada e esta por sua vez será decryptografada com a chave pública do remetente e finalmente obter a mensagem original.

A combinação das duas criptografias, a com chave privada do remetente e com a chave pública do receptor, garante tanto que quem vai receber a mensagem é realmente o destinatário correto como garante que o remetente é quem ele diz ser (STALLINGS, 2014), assim como demonstra a Figura 7.

Figura 7 – Fluxo da mensagem criptografada quando o foco é confidencialidade e autenticidade



Fonte:

## 2.3 Cifra de blocos

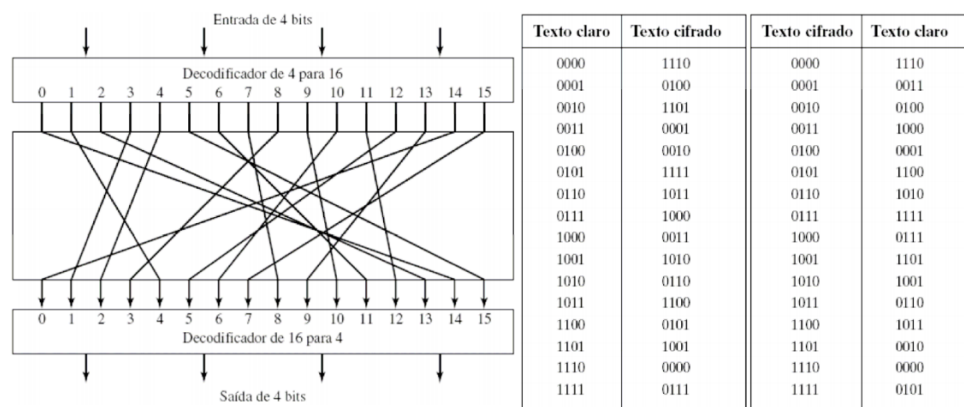
A cifra de blocos manipula um bloco da mensagem clara como um todo e utiliza este para criar um bloco de mensagem cifrada de mesmo tamanho. É comum a utilização de um bloco com 64 ou 128 bits. Uma cifra de bloco pode ser usada para alcançar o mesmo efeito que uma cifra de fluxo. Cifras de fluxo trata de certo fluxo de dados singularmente, bit a bit ou byte a byte. São implementações clássicas das cifras de fluxo as cifras de Vigenère auto chaveada e a cifra de Vernam e a mais utilizada atualmente é o RC4.

A maioria das criptografias simétricas voltadas para ambientes de rede de computadores são implementadas com cifra de blocos e grande parte dos algoritmos de cifra de bloco simétrico usados na atualidade são baseados na cifra de blocos de Feistel.

A cifra de bloco trabalha com um bloco de dados da mensagem clara de  $n$  bits para produzir um bloco de mensagem cifrada de também de  $n$  bits. Se a cifra de bloco se utilizar de um  $n$  baixo então a cifra será equivalente a uma cifra de substituição simples. E se  $n$  for grande o suficiente e ainda assim for permitida a substituição reversível entre os blocos de mensagem cifrada e não cifrada, então a mensagem clara estaria tão camuflada que a criptoanálise se tornaria inviável.

Uma cifra de substituição reversível qualquer, Feistel chamava esse cenário de cifra de bloco ideal, para um grande tamanho de bloco não é viável, do ponto de vista de desempenho e implementação, assim como demonstra a Figura 8.

Figura 8 – Cifra de bloco ideal



Fonte:

*Feistel* propunha que era possível chegar mais próximo da cifra de bloco ideal. Era só utilizar uma cifra de produto, ou seja, a execução de duas ou mais cifras em sequência, tornando assim o produto final criptograficamente mais forte do que se poderia obter através de qualquer uma das cifras componentes do produto. A estrutura de Feistel consiste em repetidas rodadas do mesmo procedimento. A cada vez que o procedimento se repete é realizada a substituição em metade dos dados da mensagem sendo processados, logo após



se permuta as duas metades. A chave sendo utilizada é expandida de modo que uma chave diferente seja utilizada em cada iteração.

Na prática *Fiestel* propôs o uso de uma cifra que alternava entre substituições e permutações, o que na realidade é uma implementação prática do que *Claude Shannon* já havia proposto como, cifra de produto que alterne confusão e difusão (STALLINGS, 2014).

## 2.4 DES

Foi estabelecido pela IBM no final da década de 1960 um projeto de pesquisa sobre criptografia de computadores que trazia a sua frente Horst Feistel. Com a conclusão do projeto em 1971 foi apresentado como resultado um algoritmo denominado LUCIFER, que foi comercializado ao Lloyd's localizado em Londres para ser utilizado em um sistema de caixa eletrônico que também havia sido desenvolvido pela IBM. LUCIFER é uma cifra de bloco de Feistel que operava com blocos de 64 bits e se utilizava de uma palavra com tamanho de 128 bits.

Com o sucesso do LUCIFER a IBM mobilizou um novo projeto, dessa vez com o intuito de tornar o LUCIFER comercializável em grande escala, conseguindo coloca-lo em um único chip. Este projeto, que por sua vez era liderado por Walter Tuchman e Carl Meyer, contava com a ajuda tanto de consultores externos como orientação técnica da NSA. O resultado deste novo projeto foi um LUCIFER mais refinado e mais resistente à criptoanálise, mas continha um tamanho de chave de 56 bits, para poder caber em um chip.

A National Bureau of Standards (NBS), em 1973, solicitou propostas de uma cifra para ser padronizada a nível nacional. A IBM enviou o LUCIFER refinado, o que continha 56 bits, e por ser o melhor algoritmo proposto foi, em 1977, adotado como *Data Encryption Standard* (DES).

Com uma chave de 56 bits, existem ao todo 256 possíveis chaves, algo próximo de  $7,2 \times 10^{16}$  chaves. Aparentemente, um número tão grande de possibilidades é algo praticamente impossível de se descobrir se utilizando o método de força bruta. Por exemplo, uma única máquina processando uma criptografia DES por microssegundo levaria mais de mil anos para quebrar a cifra.

Por mais que para as máquinas de hoje, uma criptografia DES a cada microssegundo pareça irreal, pois a velocidade das máquinas de hoje já é bem superior as máquinas da década de 70, para aquela época não era algo tão simples. Mesmo assim, em 1977, Diffie e Hellman declararam que já existia tecnologia suficiente para se criar uma máquina que concentraria, de forma paralela, um milhão de dispositivos criptográficos, cada um com o poder de processamento da máquina citada no exemplo anterior. Isso reduziria o tempo de



quebra da cifra para cerca de dez horas. Infelizmente, tal configuração na época custaria em torno de vinte milhões de dólares.

Finalmente, em 1998, vinte e um anos depois, o DES provou ser inseguro, quando a Eletronic Frontier Foundation (EFF) anunciou que tinha quebrado uma cifra DES utilizando uma máquina “decifradora de DES” montada por menos de 250 mil dólares. Como se isso já não fosse suficiente para tornar o DES um algoritmo criptográfico ultrapassado, ainda existe a constante evolução do hardware, aumentando cada vez mais a velocidade de processamento e consequentemente possibilitando tanto que o método da força bruta seja mais viável, temporalmente falando, como também que sejam criados algoritmos de criptografia que se utilizem melhor desse novo poder de processamento.

Felizmente já existem várias alternativas para o DES, entre elas estão o Advanced Encryption Standard (AES) e o Triple DES (3DES) (STALLINGS, 2014).

## 2.5 3DES

A criptografia múltipla é quando um algoritmo de criptografia é utilizado repetidas vezes. Primeiramente, a mensagem clara é criptografada utilizando um algoritmo de criptografia. A mensagem cifrada é então usada como entrada para um algoritmo de criptografia, podendo ser o mesmo utilizado anteriormente ou algum outro algoritmo, e esse processo pode ser repetir por indefinidas vezes.

O Triple DES (correlacionar com 3DES) é um exemplo de criptografia múltipla. Ele se utiliza do algoritmo DES três vezes, usando duas ou três chaves diferentes.

Em seu estado inicial a criptografia múltipla possui dois estágios, no caso do DES duplo, cada um se utilizando de uma chave diferente uma da outra.

No caso do DES duplo a criptografia E de uma mensagem M se utilizando de duas chaves K1 e K2 resultando em um texto criptografado C seria assim:

$$C = E(K_2, E(K_1, M))$$

Já a sua decriptografia D seria assim, aplicando as chaves inversamente:

$$M = D(K_1, D(K_2, C))$$

O DES triplo veio como uma alternativa clara para sanar o problema do DES simples, gerado pelo avanço computacional, uma vez que aumenta o custo do ataque da mensagem clara conhecida para 2112 (quando se utiliza de duas chaves) o que está além do possível, pelo menos, atualmente. Mas se utilizar de três estágios com três chaves diferentes exige um tamanho de chave de 168 bits (56 x 3), o que pode ser custoso.

Para diminuir o problema citado, Tuchman propôs uma alternativa se utilizando somente de duas chaves:

$$C = E(K_1, D(K_2, E(K_1, M)))$$

A solução acima apresentada se tornou relativamente popular tendo sido adotada para uso nos padrões de gerenciamento de chaves ANS X9.17 e ISO 8732.

Atualmente contra o 3DES não existem ataques criptoanalíticos práticos, visto que, o custo de uma pesquisa de chave por força bruta seria de ordem  $2^{112}$  (quando fossem utilizadas somente duas chaves) o que é equivalente a aproximadamente  $5 \times 10^{33}$ .

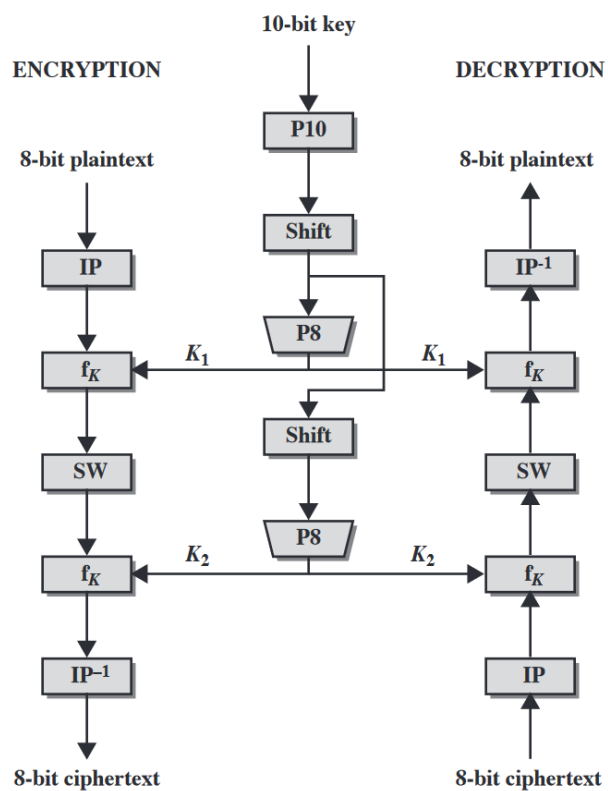
Mesmo o 3DES com duas chaves sendo de difícil ataque, ainda pode haver uma certa preocupação. Portanto pesquisadores creem que o melhor seria se utilizar do 3DES com três chaves. Como já dito anteriormente o 3DES com três chaves possui uma chave de tamanho efetivo de 168 bits, o que a torna mais segura em relação com o de duas chaves, e possui a seguinte forma (STALLINGS, 2014):

$$C = E(K_3, D(K_2, E(K_1, M)))$$

## 2.6 S-DES

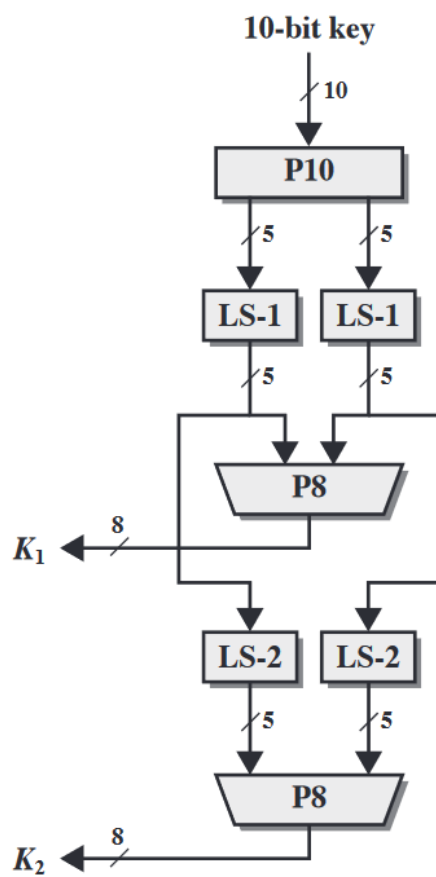
O *Simplified Data Encryption Standard* (S-DES) é uma versão simplificada do algoritmo *Data Encryption Standard* (DES).

Ele se utiliza de parâmetros de entrada menores que os possíveis com o DES e faz somente 2 permutações, tornando assim este o melhor candidato para análise quando o objetivo é aprendizado.

Figura 9 – S-DES *scheme*

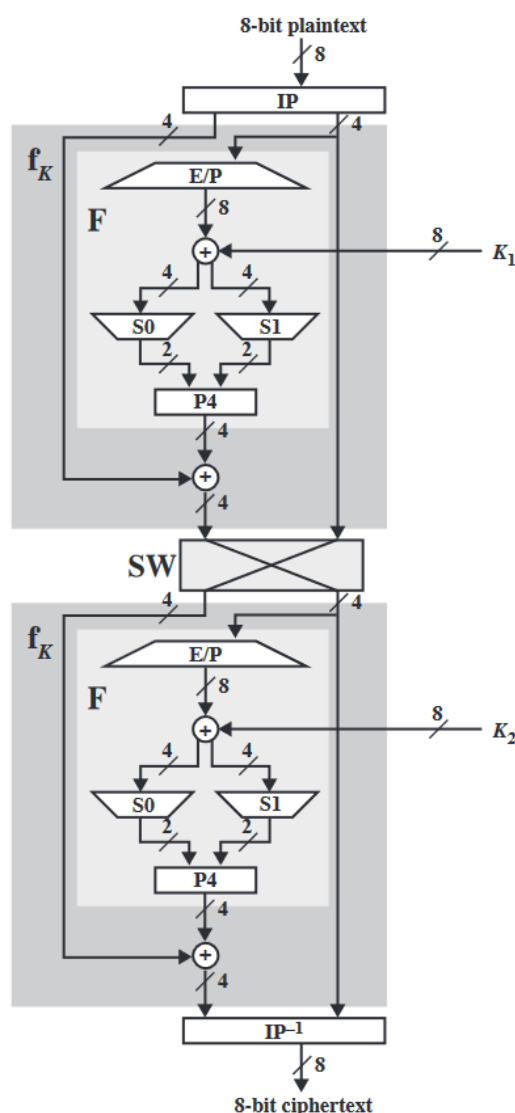
Fonte: "Figura G.1 do apêndice G"(STALLINGS, 2010)

Figura 10 – Geração das Chaves do S-DES



Fonte: "Figura G.2 do apêndice G"(STALLINGS, 2010)

Figura 11 – Detalhamento da criptografia do S-DES



Fonte: "Figura G.3 do apêndice G"(STALLINGS, 2010)

Continuar...

## 2.7 Ferramentas de apoio ao ensino

A educação abrange dar e receber conhecimento. Qualquer sociedade possui esse cenário, que é responsável pela manutenção e propagação às gerações que se seguem, da cultura necessária à convivência de um membro na sua sociedade (HAMAWAKI; PELEGRINI, 2009).

Dentre as formas de aprendizagem, são citadas duas teorias importantes por Ausubel (AUSUBEL; NOVAK; HANESIAN, 1980), a aprendizagem significativa e a mecânica.

A primeira diz respeito à ideia de que a aprendizagem é um processo por onde

uma informação recém-adquirida relaciona-se com um aspecto importante da estrutura de conhecimento do aluno, ou seja, quando a nova informação adquirida vincula-se em conhecimentos relevantes previamente adquiridos na estrutura cognitiva do aluno. A aprendizagem significativa é onde o aluno realmente aprende.

A aprendizagem mecânica por sua vez ocorre quando o novo conhecimento se associa pouco ou não se associa a algum conhecimento importante já existente na estrutura cognitiva do aprendiz. Quando isso ocorre o conhecimento adquirido recentemente é gravado de maneira arbitrária. Essa aprendizagem é inviável quando um aluno recebe um novo conhecimento em uma área de conhecimentos nova para ele. Ou seja, aprendizagem mecânica acontece somente até que alguns elementos do conhecimento pré-adquirido, relevantes a novas informações na mesma área de conhecimento, existam na estrutura cognitiva do aluno e possam ser utilizadas como base para a aprendizagem significativa. É através dessa aprendizagem que se inicia a criação, na cabeça do aluno, de uma estrutura cognitiva mais complexa, onde ele deixa de aprender de forma passiva e mecânica e passa a assimilar o conteúdo de forma mais clara e não linear (AUSUBEL; NOVAK; HANESIAN, 1980).

Existe um problema comum entre professores e alunos de disciplinas do curso de Ciência da Computação, e este é: Existe uma dificuldade em demonstrar a real dinâmica dos eventos computacionais. Por melhor que o mestre possua conhecimento e comunicação, que o aluno dedique raciocínio e atenção, a total compreensão dos conceitos abordados fica comprometida pela forma de abordagem das disciplinas.

O computador não deve ser o substituto do professor, mas sim, tomar o papel de ferramenta educacional auxiliando o aprendizado. Com isso o papel do professor passa a ser mais produtivo.

Primeiramente, ferramentas voltadas ao ensino, já se utilizando do computador, se baseavam na máquina de B. F. Skinner, onde se utilizava a ideia de instrução programada, que consistia de dividir o conteúdo total em pequenas partes lógicas e sequenciais.

Somente durante a década de 1960, foi dado início ao conceito de Computer Aided Instruction (CAI), ou seja, Programas Educacionais por computador (PEC). A real dispersão do CAI só aconteceu com popularização dos microcomputadores, possibilitando assim, a criação de diversos tipos de ferramentas de ensino, como, tutoriais, exercício-e-prática, avaliações, jogos, e simulações (HAMAWAKI; PELEGRINI, 2009).

## 2.8 Simuladores voltados ao ensino

Algumas situações são tão singulares, que se preparar para tais torna-se difícil. E portanto, com o intuito de proporcionar a alguém um conhecimento maior sobre estas

situações os simuladores foram criados.

A simulação é uma representação do mundo real. As primeiras simulações foram criadas com o intuito de disponibilizar um ambiente seguro para situações que envolvessem risco ao humano. Dentre elas podemos citar simulações de viagens e mergulhos profundos. Posteriormente, foram criadas com o intuito de se obter uma economia, seja ela de tempo e/ou dinheiro, que é o caso da indústria automobilística e aviação.

Diversas são as áreas de conhecimento que fazem uso das simulações (BANKS *et al.*, 2009). Na ciência da computação há uma disponibilidade de simuladores, cada um atendendo a uma respectiva disciplina, assim como redes de computadores, arquitetura de computadores, técnicas de programação e sistemas operacionais.

Os simuladores possuem um potencial bem maior do que as outras ferramentas de ensino citadas acima. A aplicação de simulações no universo acadêmico permite que o aluno desenvolva hipóteses, teste-as, analise os resultados e concretize seus conhecimentos. Simuladores devem ser utilizados como ferramenta complementar as aulas lecionadas pelo professor.

Mesmo com todas as vantagens envolvidas, o desenvolvimento de um simulador não é algo trivial. Uma vez que a ferramenta possui um cunho pedagógico, a utilização de recursos multimídia para tornar a simulação mais fiel à realidade é muito importante, e a utilização destes recursos não é algo de simples desenvolvimento.

Quando o aluno possui pouco conhecimento na área, os simuladores são indicados como melhor ferramenta de auxílio. Principalmente, se forem para auxiliar cursos de extensão e graduação (MAIA, 2001) (MAIA; PACHECO, 2003).

Até o final da escrita desse documento o único simulador de criptografia gratuito que foi encontrado foi o Enigma Simulator by Terry Long (<http://www.terrylong.org/>), e este, se concentra apenas na simulação do algoritmo da máquina Enigma utilizada na Alemanha no período da segunda guerra mundial.

## 3 Ferramenta desenvolvida

Nesse capítulo descrevo o **CryptoEdu - simulador de algoritmos de criptografia com finalidade educacional** desenvolvido no decorrer da escrita desse trabalho de conclusão de curso.

O simulador permite a execução completa e passo a passo de todas as etapas envolvidas tanto na criptografia quando na descryptografia utilizando o algoritmo *Simplified Data Encryption Standard* (S-DES).

### 3.1 Informações técnicas

O simulador foi desenvolvido na linguagem *Javascript* utilizando *React.js* e *Typescript*. O ambiente de desenvolvimento utilizado foi o *Visual Studio Code*. O código fonte do simulador é *open source* e está disponibilizado no *GitHub* no endereço <https://github.com/TanielianVB/CryptoEdu/>.

No *GitHub* é possível: Gerenciar problemas encontrados e melhorias sugeridas através de *Issues*.; Facilmente integrar melhorias implementados por terceiros através de *Pull Requests*.; A publicações de novas versões está automatizado e reagindo à publicação de qualquer nova versão no repositório através da ferramenta *Netlify* (<https://www.netlify.com/>). Esses *deploys* podem ser visualizados em <https://app.netlify.com/sites/cryptoedu/deploys/>.; E até, caso necessário, criar uma nova versão do simulador de propriedade de terceiros através do *Fork*.

O simulador está disponibilizado ao usuário através da internet no endereço <https://cryptoedu.netlify.app/>. O simulador pode ser visualizado em qualquer *browser*, inclusive em dispositivos móveis. No entanto a melhor experiência é obtida quando se utiliza *desktop*.

### 3.2 Usuário

O público alvo da ferramenta são os alunos de Computação, mas os atores desta não estão limitados à. Todos que desejam aprender como a criptografia de cifra de blocos funciona, refinar e/ou lapidar os conhecimentos já adquiridos ou até somente desmistificar esse conteúdo complexo chamado criptografia.



### 3.3 Estrutura da interface

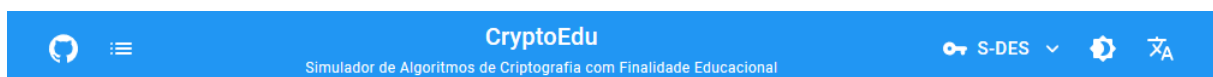
Nessa seção descrevo os elementos contidos na interface de maneira estrutural.

Figura 12 – Estrutura da interface



#### 3.3.1 Cabeçalho

Figura 13 – Cabeçalho



A região do cabeçalho possui:

- Link do repositório onde está localizado tanto o código fonte do simulador como um pdf do TCC: <https://github.com/TanielianVB/CryptoEdu>
- Link do questionário de *feedback* de utilização do simulador.
- O nome do simulador: **CryptoEdu - Simulador de Algoritmos de Criptografia com Finalidade Educacional**.
- *Combo Box* de escolha do algoritmo que está em execução.
- Botão para alternar entre o tema **claro** e **escuro**.
- Botão para alterar o idioma no qual o simulador está sendo exibido.

### 3.3.2 Conteúdo

Figura 14 – Conteúdo

A região principal do site será onde cada passo da execução do algoritmo selecionado irá ocorrer. Em cada passo serão listados as etapas executadas em cada passo.

Inicialmente irá conter uma interface percorrendo sobre o algoritmo selecionado e posteriormente cada passo que estará sendo executado. É possível navegar por esses passos através dos botões de navegação (Próximo e Anterior) exibidos na parte inferior da região de cada passo. No último passo o botão Próximo é substituído pelo botão Reiniciar para facilitar o início de uma nova execução.

### 3.3.3 Rodapé

Figura 15 – Rodapé



A região do rodapé contém:

- Todos os passos necessários para a execução do algoritmo em execução. Ao se passar o mouse sobre eles é exibida uma *tooltip* com um nome mais explicativo do passo.
- Quais passos já foram completados.
- Qual passo o usuário se encontra no momento.
- Quais passos ainda faltam ser completados para finalizar a execução do algoritmo.

É possível também, a qualquer momento, navegar para qualquer passo diretamente. Basta-se clicar sobre este.

### 3.3.4 Mobile

Figura 16 – Visualização em um *browser* de dispositivo móvel

**CryptoEdu** S-DES

Simulador de Algoritmos de Criptografia com Finalidade Educacional

## S-DES - Simplified Data Encryption Standard

**CRIPTOGRAFAR** DESCRIPTOGRAFAR

Mensagem \*

01110010

1 char ou 8 bits

Bits da mensagem:

1	2	3	4	5	6	7	8
0	1	1	1	0	0	1	0

= Char r

Chave \*

1010000010

10 bits

Bits da chave:

1	2	3	4	5	6	7	8	9	10
1	0	1	0	0	0	0	0	1	0

☐ Iniciar regiões de explicação abertas

**PRÓXIMO >**

Progress bar: 1 — 2 — 3 — 4 — 5 — 6 — 7 — 8

I — C — IP —  $f_{K_1}$  — SW —  $f_{K_2}$  —  $IP^{-1}$  — F

Embora não seja o ideal, o simulador se comporta bem em dispositivos móveis melhorando ainda mais a sua acessibilidade.

## 3.4 Interface do conteúdo

Descrição da interface apresentada em cada um dos passos da execução do algoritmo.

Cada passo possui no mínimo uma etapa e cada etapa possui uma região de explicação onde a etapa é descrita de maneira que possibilite o usuário compreender como a etapa é executada, como essa deve ser compreendida, como esta difere de outras etapas similares e qual a definição matemática desta etapa, caso haja.

A explicação pode ser acessada através do clique do botão interrogação à direita do cabeçalho que contém o título da referida etapa. Esse clique irá expandir a região que contém a explicação da etapa. É possível também que todas as regiões de explicação já iniciem abertas. Basta habilitar a opção **Iniciar regiões de explicação abertas** ao iniciar a execução da criptografia ou descriptografia.

### 3.4.1 Tela inicial - entrada de dados

Figura 17 – Tela inicial - entrada de dados

The screenshot shows the 'S-DES - Simplified Data Encryption Standard' interface. At the top, there are two tabs: 'CRİPTOGRAFAR' (selected) and 'DESCRIPTOGRAFAR'. Below the tabs, there are two input fields: 'Mensagem \*' with a value of '01110010' and 'Chave \*' with a value of '1010000010'. Below the message field, it says '1 char ou 8 bits' and 'Bits da mensagem:'. Below the key field, it says '10 bits' and 'Bits da chave:'. Below these, there are two rows of bit boxes. The first row shows the message bits: 1 (0), 2 (1), 3 (1), 4 (1), 5 (0), 6 (0), 7 (1), 8 (0), followed by an equals sign and a box labeled 'Char' containing 'r'. The second row shows the key bits: 1 (1), 2 (0), 3 (1), 4 (0), 5 (0), 6 (0), 7 (0), 8 (0), 9 (1), 10 (0). At the bottom, there is a toggle switch labeled 'Iniciar regiões de explicação abertas' which is currently turned off. In the bottom right corner, there is a blue button labeled 'PRÓXIMO >'. A red question mark icon is in the top right corner.

A interface inicial faz uma breve descrição do algoritmo selecionado na *Combo Box* de escolha de algoritmo buscando situar o usuário no contexto selecionado para execução. No momento só será disponibilizado um algoritmo: *Simplified Data Encryption Standard* (S-DES).

O usuário pode então escolher se ele deseja o fluxo de execução **criptografar** ou **descriptografar** e assim então informar valores customizados para os campos **Mensagem** (**Mensagem cifrada** caso o fluxo escolhido tenha sido o **descriptografar**) e **Chave**.

Os algoritmos recebem como entrada bits. Por conta disso, são exibidos os bits contidos nos campos e que serão utilizados para a execução do fluxo escolhido. Tanto para facilitar o preenchimento do campo **Mensagem** como para melhorar a compreensão do usuário sobre o valor contido no campo, é possível informar no campo uma letra, visto que essa pode, e é, facilmente convertida para 8 bits. Isso ajuda não somente no preenchimento do campo durante a execução do fluxo **criptografar** mas também na validação do resultado obtido da execução do fluxo **descriptografar** visto que é possível, mais facilmente, comparar as mensagens (tanto a de entrada do fluxo **criptografar** quando a de saída do fluxo **descriptografar**) se estas forem letras. Exemplo: A letra 'T' gera a sequência de bits: **01010100**.

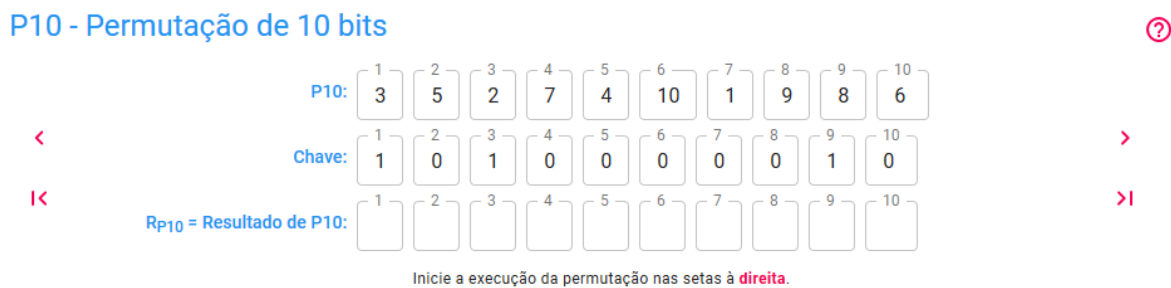
Nessa tela também é possível habilitar a opção **Iniciar regiões de explicação abertas** que fará com que, durante a navegação pelos passos da execução do algoritmo todas as regiões de explicação já se iniciem abertas. Caso o usuário não queira mais visualizar a explicação para alguma etapa. Ele pode esconder a explicação clicando no botão à direita do título da etapa.

### 3.4.2 Passo Chaves - Geração das chaves

Nesse passo estão concentradas todas as etapas necessárias para a geração das chaves  $K_1$  e  $K_2$ . As etapas contidas nesse passo estavam inicialmente divididas em 3 passos. Mas por ter gerado dúvida sobre como essas etapas se relacionavam tanto entre elas como com os outros passos da execução do algoritmo estas etapas estão em um mesmo passo.

#### 3.4.2.1 Etapa P10 - Permutação de 10 bits

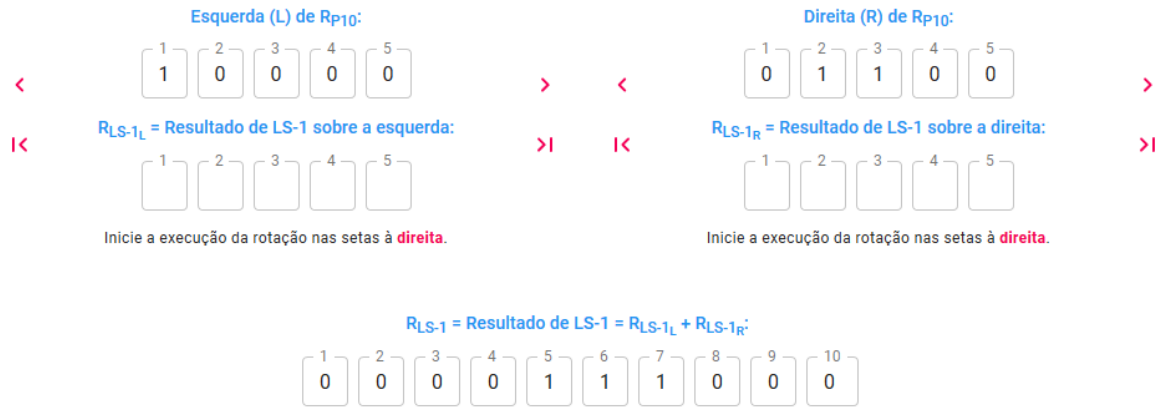
Figura 18 – Etapa P10 - Permutação de 10 bits



A interface da etapa P10 contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P10 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P10. O parâmetro de entrada da permutação P10 é a Chave recebida da tela de entrada de dados. O resultado dessa execução será o parâmetro de entrada para a próxima etapa, LS-1.

3.4.2.2 Etapa LS-1 - *Circular Left Shift* de 1 posiçãoFigura 19 – Etapa LS-1 - *Circular Left Shift* de 1 posição

## LS-1 - Circular Left Shift de 1 posição

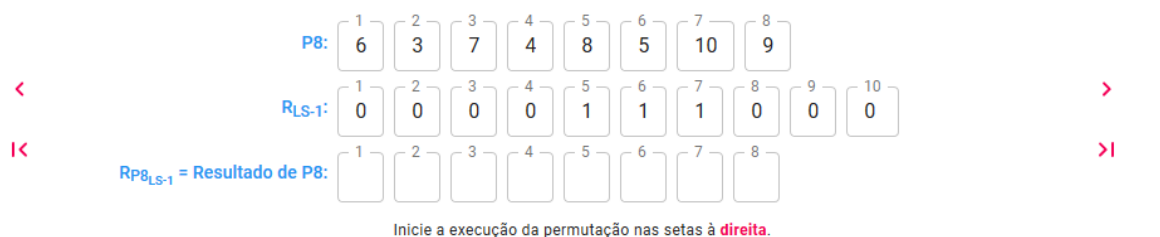


A interface da etapa LS-1 contextualiza o objetivo pelo qual essa etapa existe e explica como esta etapa deve ocorrer, incluindo a divisão na metade do valor obtido na etapa anterior ( $P_{10}$ ) e o que é o processo de rotação. Após tal contextualização se exibe dois componentes capazes de executar passo a passo a rotação circular para a esquerda de 1 posição. Cada um destes componentes irá rotacionar uma das metades. O resultado desta etapa é então a junção das metades após suas rotações individuais.

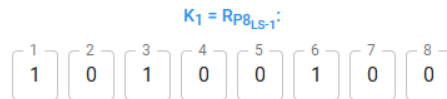
## 3.4.2.3 Etapa P8 - Permutação de 8 bits sobre o resultado de LS-1

Figura 20 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-1

## P8 - Permutação de 8 bits



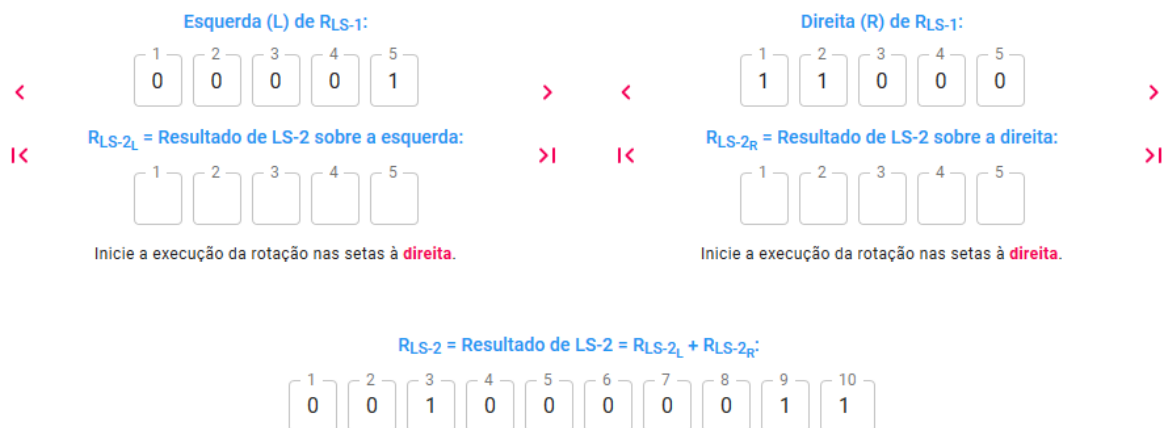
A interface da etapa P8 contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P8 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P8. O parâmetro de entrada da permutação P8 é o resultado obtido na etapa anterior, LS-1.

3.4.2.4 Resultado Chave  $K_1$ Figura 21 – Chave  $K_1$  - Geração da primeira chave  $K_1$  $K_1$  - Primeira chave

O resultado da aplicação da função de permutação P8 sobre o resultado da etapa LS-1 será a primeira chave  $K_1$ . Esta será utilizada em uma etapa futura durante a criptografia ou descryptografia da mensagem.

3.4.2.5 Etapa LS-2 - *Circular Left Shift* de 2 posiçõesFigura 22 – Etapa LS-2 - *Circular Left Shift* de 2 posições

## LS-2 - Circular Left Shift de 2 posições

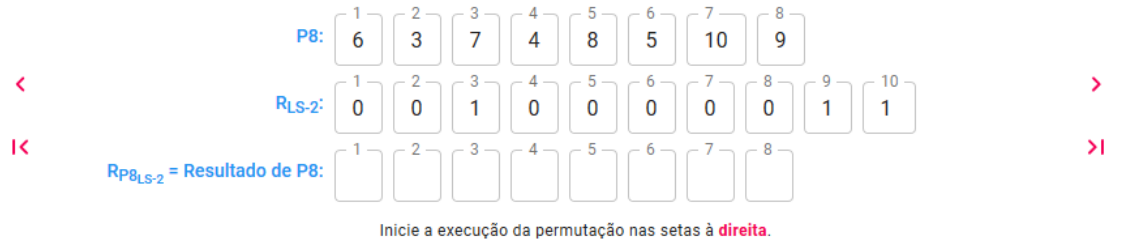


A interface da etapa LS-2 contextualiza o objetivo pelo qual essa etapa existe e explica como esta etapa deve ocorrer, incluindo a divisão na metade do valor obtido na etapa LS-1 e o que é o processo de rotação. Após tal contextualização se exhibe dois componentes capazes de executar passo a passo a rotação circular para a esquerda de 2 posições. Cada um destes componentes irá rotacionar uma das metades. O resultado desta etapa é então a junção das metades após suas rotações individuais.

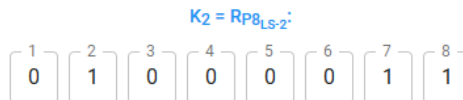
## 3.4.2.6 Etapa P8 - Permutação de 8 bits sobre o resultado de LS-2

Figura 23 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-2

## P8 - Permutação de 8 bits



A interface da etapa P8 contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P8 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P8. O parâmetro de entrada da permutação P8 é o resultado obtido na etapa anterior, LS-2.

3.4.2.7 Resultado Chave  $K_2$ Figura 24 – Chave  $K_2$  - Geração da segunda chave  $K_2$  $K_2$  - Segunda chave

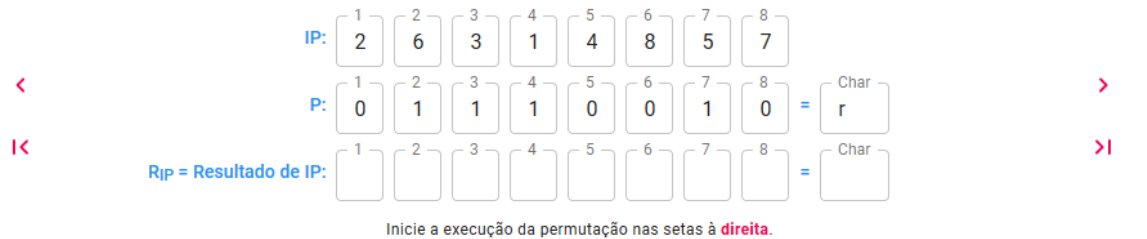
O resultado da aplicação da função de permutação P8 sobre o resultado da etapa LS-2 será a segunda chave  $K_2$ . Esta será utilizada em uma etapa futura durante a criptografia ou descryptografia da mensagem.



### 3.4.3 Passo IP - Permutação Inicial

Figura 25 – Passo IP (*Initial Permutation*) - Permutação Inicial

#### IP - Permutação Inicial



A interface da etapa IP contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação IP (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação IP. O parâmetro de entrada da permutação IP é a Mensagem (*Plaintext* P) recebida da tela de entrada de dados.

Figura 26 – L (esquerda) & R (direita)

#### L (esquerda) & R (direita)



O resultado desse passo é a divisão do resultado obtido da permutação IP em duas metades, L (*left*) e R (*right*), que serão enfim passados por parâmetro para a execução da função  $f_K$ .

### 3.4.4 Passo $f_K$ - Função que usa uma chave

Figura 27 – Passo  $f_K$  - Função que usa uma chave

#### $f_{K_1}$ - Função que usa a chave $K_1$

A função  $f_K$  é o componente mais complexo da execução do algoritmo e consiste de uma combinação de permutações e substituições e será chamada duas vezes durante o fluxo de execução, sendo uma vez para cada chave ( $K_1$  e  $K_2$ ). Como estamos criptografando, a primeira execução da função  $f_K$  deverá se utilizar da chave  $K_1$ . A função  $f_K$  é definida por:

$$f_K(L, R) = (L \oplus F(R, SK), R)$$

A função  $f_K$  se utiliza da função F que por sua vez é definida por uma sequencia de passos. Vamos executar iniciando da função mais interna até a mais externa e analisar cada parte.

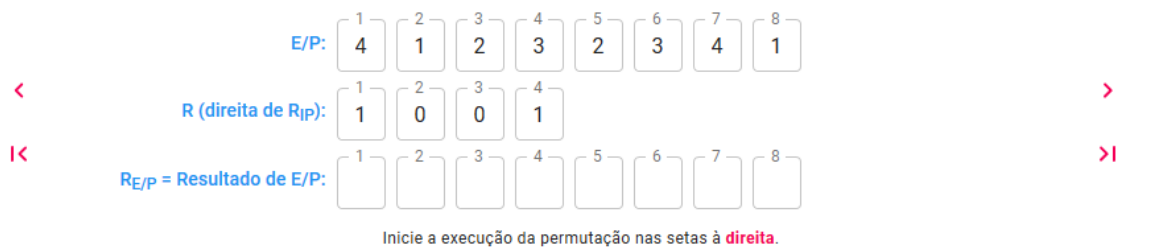
Esse passo é o único passo que se repete durante a execução da criptografia ou descryptografia. Essa função recebe 2 parâmetros de entrada. Um é uma sequência de 8 bits divididos entre L & R (4 bits na esquerda L e 4 bits na direita R) e o segundo parâmetro é a chave. Caso a execução escolhida seja a criptografia, a primeira vez que essa função é executada a chave utilizada será a  $K_1$  e na segunda vez será a  $K_2$ . Caso a execução escolhida seja a descryptografia, a ordem de utilização das chaves será inversa. A interface explica a tais peculiaridades e apresenta a função matemática desta.

Nesse passo estão concentradas todas as etapas da execução da função  $f_K$ . As etapas contidas nesse passo estavam inicialmente divididas em 2 passos. Um indo até o primeiro XOR e outro indo até o segundo XOR. Mas por esse passo se repetir optou-se por deixar todas as etapas em um único passo.

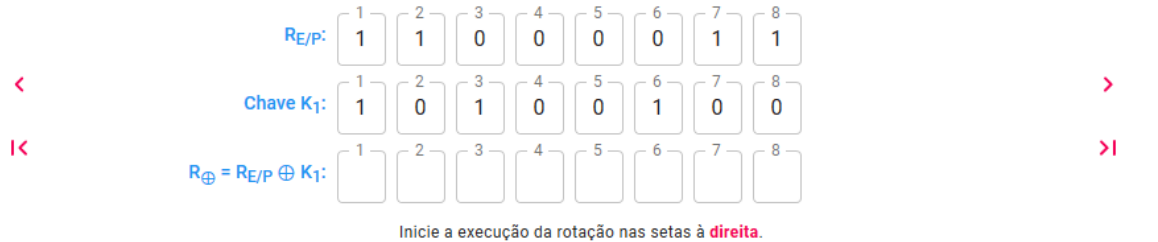
#### 3.4.4.1 Etapa E/P - Permutação de Expansão

Figura 28 – Etapa E/P - Permutação de Expansão

##### E/P - Permutação de Expansão



A interface da etapa E/P contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação E/P (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação E/P. O parâmetro de entrada da permutação E/P é o parâmetro de entrada R da função  $f_K$  (4 bits do lado direito dos 8 bits recebidos de entrada).

3.4.4.2 Etapa XOR - OU exclusivo com  $K_1$ Figura 29 – Etapa XOR - OU exclusivo com  $K_1$ XOR - OU exclusivo -  $\oplus$ 

A interface da primeira etapa XOR descreve o que ocorre nessa etapa e exibe um componente capaz de executar bit a bit a operação XOR (OU exclusivo) para obtenção do resultado dessa etapa. Os parâmetros de entrada dessa etapa são: 1. O resultado da permutação de expansão E/P e 2. Ou a chave  $K_1$  ou a chave  $K_2$ . Caso a execução escolhida seja a criptografia, a primeira vez que essa etapa é executada a chave utilizada será a  $K_1$  e na segunda vez será a  $K_2$ . Caso a execução escolhida seja a descriptografia, a ordem de utilização das chaves será inversa. A interface reflete essas particularidades em cada fluxo de execução.

## 3.4.4.3 Etapa S0 &amp; S1 - Substituições S0 e S1

Figura 30 – Etapa S0 &amp; S1 - Substituições S0 e S1

## S0 &amp; S1 - Substituições S0 e S1

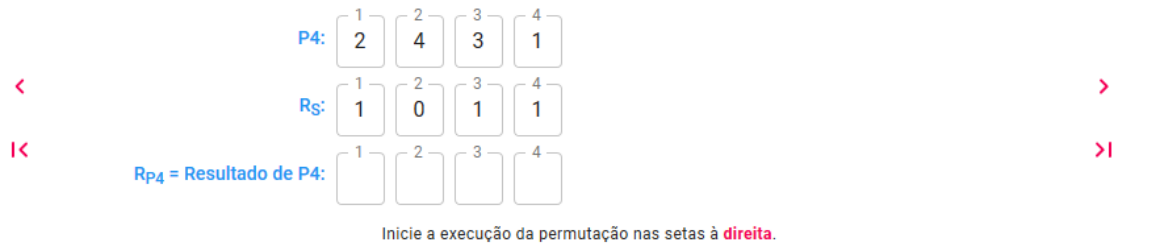


A interface da etapa das substituições S0 e S1 contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma substituição, apresenta a obtenção dos parâmetros de entrada para cada substituição e explica como uma substituição deve ser interpretada. Após tal contextualização se exhibe, para cada substituição, um componente capaz de executar passo a passo a substituição. Os parâmetros de entrada das substituições S0 e S1 são, respectivamente, as metades esquerda e direita do resultado obtido na etapa anterior (XOR com  $K_1$ ). O resultado desta etapa é então a junção das metades após suas substituições individuais.

## 3.4.4.4 Etapa P4 - Permutação de 4 bits

Figura 31 – Etapa P4 - Permutação de 4 bits

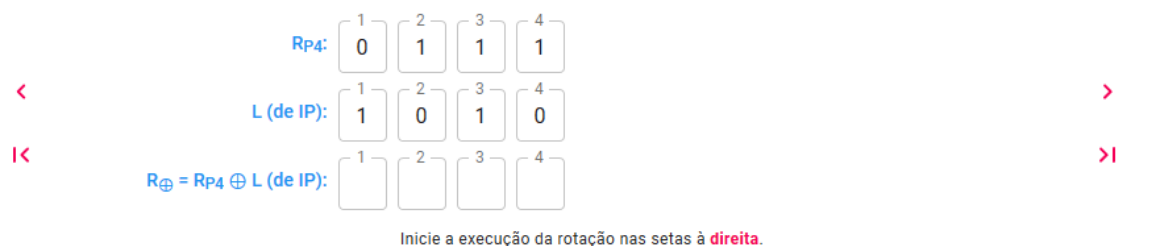
## P4 - permutação de 4 bits



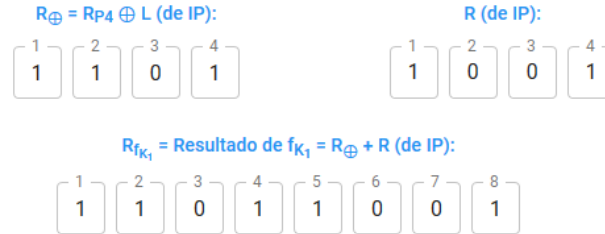
A interface da etapa P4 contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P4 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P4. O parâmetro de entrada da permutação P4 é o resultado obtido na etapa anterior (Substituições S0 e S1). O resultado dessa execução será o parâmetro de entrada para a próxima etapa, XOR com L.

## 3.4.4.5 Etapa XOR - OU exclusivo com L

Figura 32 – Etapa XOR - OU exclusivo com L

XOR - OU exclusivo -  $\oplus$ 

A interface da segunda etapa XOR descreve o que ocorre nessa etapa e exibe um componente capaz de executar bit a bit a operação XOR (OU exclusivo) para obtenção do resultado dessa etapa. Os parâmetros de entrada dessa etapa são: 1. O resultado da permutação P4 e 2. L (metade esquerda do parâmetro de entrada de  $f_K$ ). Na primeira vez que  $f_K$  é executada L será a esquerda do resultado da permutação inicial (IP) e na segunda vez L será a esquerda do resultado da Troca (SW).

3.4.4.6 Resultado de  $f_K$ Figura 33 – Resultado de  $f_K$ Resultado de  $f_{K_1}$ 

A interface do resultado de  $f_K$  exibe as metades que compõem o resultado da função e a junção destes. A metade esquerda desse resultado sempre será o resultado da etapa anterior, XOR. A metade direita será a metade direita do parâmetro de entrada da função  $f_K$ . Que na primeira vez é a metade direita do resultado da permutação inicial (IP) e na segunda vez é a metade direita da Troca (SW).

## 3.4.5 Passo SW - Troca

Figura 34 – Passo SW - Troca

## SW - Troca



A interface do passo Troca (SW) explica a definição da troca, apresenta a função de troca (incluindo função matemática) e explica como esta deve ser interpretada. Este é o passo intermediário entre as execuções das funções  $f_K$ . Ela recebe por parâmetro o resultado da primeira execução da função  $f_K$  e o resultado deste passo é o parâmetro

de entrada para a segunda execução da função  $f_K$  juntamente com a chave que não foi utilizada pela primeira execução.

### 3.4.6 Passo $f_K$ - Função que usa a outra chave

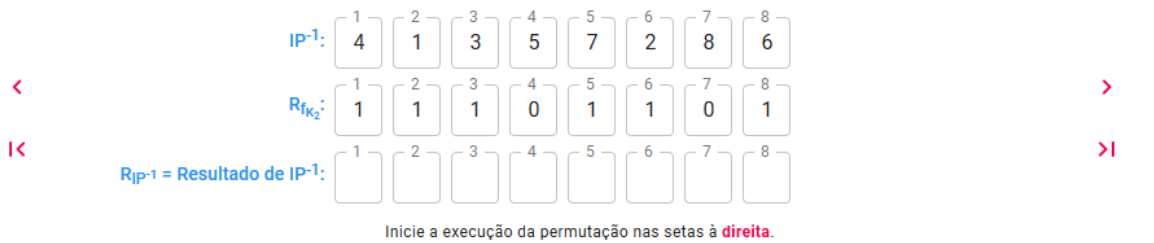
A segunda execução da função  $f_K$  é sequencialmente idêntica à primeira logo as interfaces de cada etapa são similares. As únicas diferenças são:

- O parâmetro de entrada é o resultado do passo Troca (SW) e não o resultado do passo Permutação inicial (IP) como ocorre na primeira execução da função.
- A chave utilizada é a chave que não foi utilizada na primeira execução da função  $f_K$ . Na criptografia a primeira execução da função  $f_K$  utiliza a chave  $K_1$  e a segunda execução a chave  $K_2$ . Já na descryptografia o inverso é verdade. A interface identifica o fluxo que está sendo executado e reflete essas diferenças tanto nas *labels* quanto nas explicações das etapas.

### 3.4.7 Passo $IP^{-1}$ - Permutação Inicial Inversa

Figura 35 – Passo  $IP^{-1}$  - Permutação Inicial Inversa

#### IP<sup>-1</sup> - Permutação Inicial Inversa



A interface da etapa  $IP^{-1}$  contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação  $IP^{-1}$  (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação  $IP^{-1}$ . O parâmetro de entrada da permutação  $IP^{-1}$  é o resultado obtido no passo anterior ( $f_K$ ).

### 3.4.8 Tela final - Resultado

Figura 36 – Resultado

#### Resultado da criptografia S-DES



Bits da mensagem:	<sup>1</sup> 0	<sup>2</sup> 1	<sup>3</sup> 1	<sup>4</sup> 1	<sup>5</sup> 0	<sup>6</sup> 0	<sup>7</sup> 1	<sup>8</sup> 0	=	Char
Bits da chave:	<sup>1</sup> 1	<sup>2</sup> 0	<sup>3</sup> 1	<sup>4</sup> 0	<sup>5</sup> 0	<sup>6</sup> 0	<sup>7</sup> 0	<sup>8</sup> 0	<sup>9</sup> 1	<sup>10</sup> 0
Mensagem cifrada:	<sup>1</sup> 0	<sup>2</sup> 1	<sup>3</sup> 1	<sup>4</sup> 1	<sup>5</sup> 0	<sup>6</sup> 1	<sup>7</sup> 1	<sup>8</sup> 1	=	Char
										w

A interface do último passo exibe os dois parâmetros de entrada da execução, a Mensagem e a Chave, e o resultado da execução da criptografia ou decryptografia.

Figura 37 – *Feedback request*

#### CryptoEdu



O Código fonte desse simulador está disponível no GitHub:



Peço, encarecidamente, que responda à um questionário relativo à utilização do simulador:



São somente 6 perguntas de múltipla escolha e demora, em média, 1 minuto para ser respondido.

Este simulador foi desenvolvido com o objetivo de auxiliar o ensino da criptografia. E responder ao formulário auxilia a evolução deste.

O último ponto na interface é uma solicitação do preenchimento do questionário de utilização do simulador. É exibido também um link para o repositório *GitHub* onde se encontra o simulador. No *GitHub* também é possível baixar a última versão publicada deste TCC.

## 3.5 Limites da solução

A ferramenta foi desenvolvida prevendo uma fácil extensão das suas atuais funcionalidades. Mas, visto que o escopo do projeto pode, facilmente, se exceder além do limite possível de execução de um trabalho de conclusão de curso, alguns limites foram impostos para viabilizar o desenvolvimento da ferramenta em tempo hábil. São eles:

- Somente 1 algoritmo será disponibilizado. Sendo este, o *Simplified Data Encryption Standard* (S-DES).



- Só estará disponibilizado no tema **claro**.
- Só estará disponibilizado em **Português-BR**.
- A interface terá foco para dispositivos *desktop*.

## 3.6 Comparação com outros simuladores criptográficos

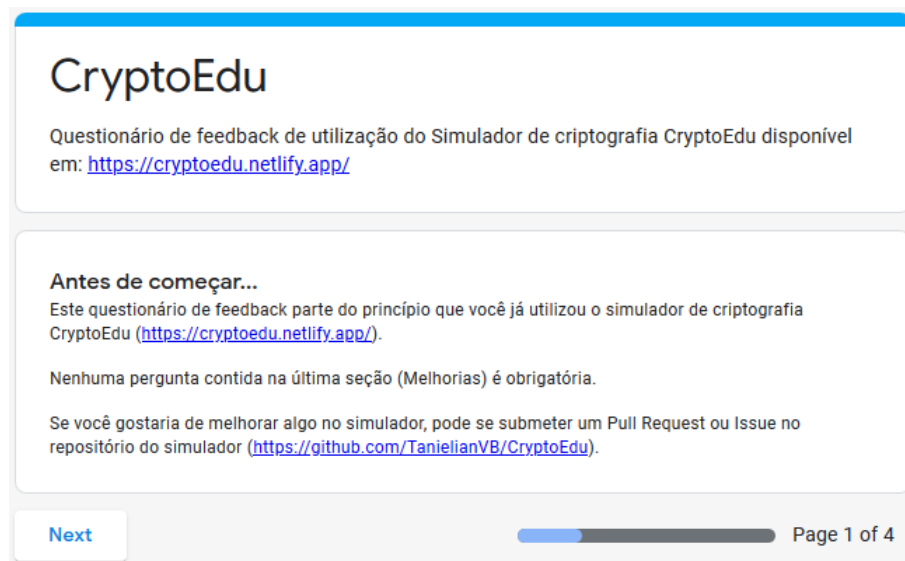
Comparando o simulador desenvolvido com todos os outros simuladores criptográficos encontrados é fácil observar que nenhum dos outros explica, de maneira tão granular, as etapas e processos envolvidos na criptografia ou descriptografia.

TODO: Extender...

## 4 Pesquisa com os alunos

O questionário de *feedback* de utilização do simulador foi disponibilizado através da utilização do *Google Forms* através do link <https://forms.gle/f1DAgWvTyM2uJyLt9>.

Figura 38 – Início do questionário



The screenshot shows the beginning of a Google Form titled 'CryptoEdu'. The header has a blue bar with the title 'CryptoEdu' in white. Below the title, the text reads: 'Questionário de feedback de utilização do Simulador de criptografia CryptoEdu disponível em: <https://cryptoedu.netlify.app/>'. The main content area has a light gray background and contains the following text: 'Antes de começar...' followed by 'Este questionário de feedback parte do princípio que você já utilizou o simulador de criptografia CryptoEdu (<https://cryptoedu.netlify.app/>).', 'Nenhuma pergunta contida na última seção (Melhorias) é obrigatória.', and 'Se você gostaria de melhorar algo no simulador, pode se submeter um Pull Request ou Issue no repositório do simulador (<https://github.com/TanielianVB/CryptoEdu>).'. At the bottom, there is a 'Next' button on the left, a progress bar in the center, and 'Page 1 of 4' on the right.

Antes de apresentar as questões para o usuário é apresentado essa tela de apresentação do questionário.

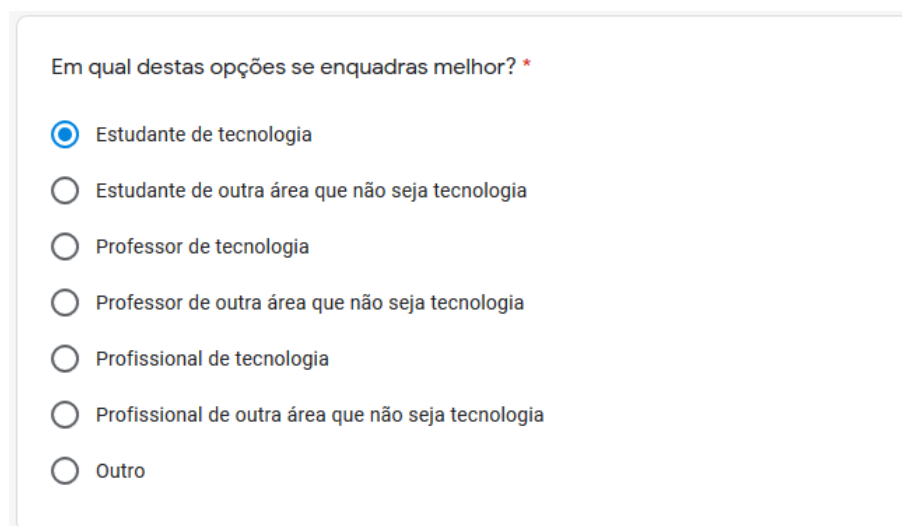
### 4.1 Perguntas

O questionário é composto de 6 perguntas, sendo 4 obrigatórias e 2 opcionais, que serão descritas à seguir.

Um fator que foi levado em consideração ao desenvolver o questionário é a dificuldade de adquirir respostas dos usuários do simulador. Tanto por esse motivo quando para facilitar a análise dos resultados se optou por se utilizar somente de perguntas de múltipla escolha.

### 4.1.1 Enquadramento do usuário

Figura 39 – Enquadramento do usuário



Em qual destas opções se enquadras melhor? \*

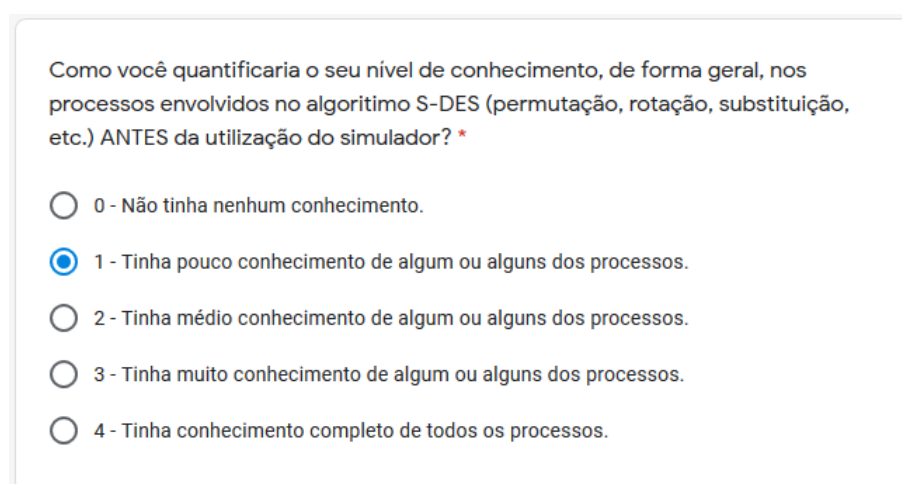
- ☒ Estudante de tecnologia
- ☐ Estudante de outra área que não seja tecnologia
- ☐ Professor de tecnologia
- ☐ Professor de outra área que não seja tecnologia
- ☐ Profissional de tecnologia
- ☐ Profissional de outra área que não seja tecnologia
- ☐ Outro

Esta pergunta tem como objetivo ter uma *baseline* do tipo de usuário que está respondendo o questionário e dessa forma ser capaz de obter uma curva de crescimento por 'perfil'.

### 4.1.2 Conhecimento antes e depois da utilização do simulador

Com as próximas 2 perguntas tem-se como objetivo ser capaz de mensurar o crescimento do nível de conhecimento do usuário nos processos presentes no algoritmo S-DES. O escopo das perguntas envolve os processos (permutação, rotação, substituição, xor e troca) presentes no algoritmo ao invés das etapas (P10, LS-1, P8, LS-2, IP, etc...) pois munido do conhecimento dos processos todas as etapas podem ser facilmente reproduzidas.

Figura 40 – Conhecimento antes da utilização do simulador

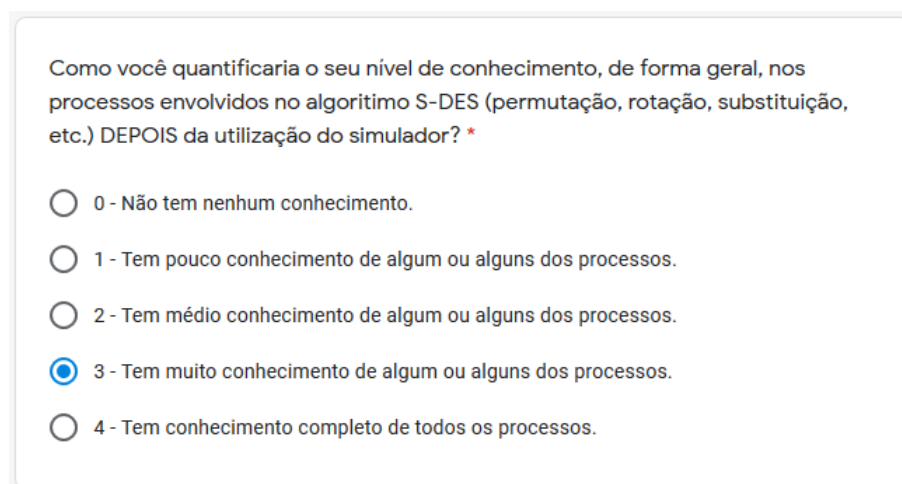


Como você quantificaria o seu nível de conhecimento, de forma geral, nos processos envolvidos no algoritmo S-DES (permutação, rotação, substituição, etc.) ANTES da utilização do simulador? \*

- ☐ 0 - Não tinha nenhum conhecimento.
- ☒ 1 - Tinha pouco conhecimento de algum ou alguns dos processos.
- ☐ 2 - Tinha médio conhecimento de algum ou alguns dos processos.
- ☐ 3 - Tinha muito conhecimento de algum ou alguns dos processos.
- ☐ 4 - Tinha conhecimento completo de todos os processos.

Para mensurar esse crescimento é questionado primeiramente como o usuário quantificaria o nível de conhecimento, de maneira geral, que ele possui nos processos presentes no algoritmo antes da utilização do simulador. Buscando obter o ponto de partida da curva de crescimento.

Figura 41 – Conhecimento depois da utilização do simulador



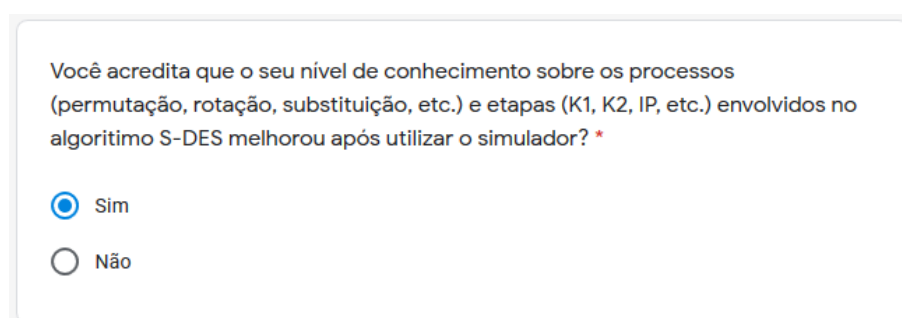
Como você quantificaria o seu nível de conhecimento, de forma geral, nos processos envolvidos no algoritmo S-DES (permutação, rotação, substituição, etc.) DEPOIS da utilização do simulador? \*

- ☐ 0 - Não tem nenhum conhecimento.
- ☐ 1 - Tem pouco conhecimento de algum ou alguns dos processos.
- ☐ 2 - Tem médio conhecimento de algum ou alguns dos processos.
- ☒ 3 - Tem muito conhecimento de algum ou alguns dos processos.
- ☐ 4 - Tem conhecimento completo de todos os processos.

A próxima pergunta indaga esse mesmo nível de conhecimento após a utilização do simulador. Buscando obter o ponto de chegada da curva de crescimento.

### 4.1.3 Efetividade do aprendizado

Figura 42 – Efetividade do aprendizado



Você acredita que o seu nível de conhecimento sobre os processos (permutação, rotação, substituição, etc.) e etapas (K1, K2, IP, etc.) envolvidos no algoritmo S-DES melhorou após utilizar o simulador? \*

- ☒ Sim
- ☐ Não

Esta pergunta tem como objetivo mensurar, de maneira absoluta, se existiu ganho de conhecimento por parte do usuário.

### 4.1.4 Melhorias no simulador

Com as próximas 2 perguntas tem-se como objetivo ser capaz de identificar quais etapas explícitas no simulador precisam ser melhoradas. O escopo das perguntas envolve as etapas (P10, LS-1, P8, LS-2, IP, etc...) presentes no algoritmo ao invés dos processos

(permutação, rotação, substituição, xor e troca) pois cada descrição e execução é singular àquela determinada etapa.

Figura 43 – Melhorias nas descrições das etapas

Quais das etapas poderiam ter sua descrição melhoradas? Marque todas aplicáveis.

- ☐ P10 - Permutação de 10 bits
- ☐ LS-1 - Rotação para a esquerda de 1 posição
- ☐ P8 - Permutação de 8 bits
- ☐ K1 - Obtenção da primeira chave
- ☐ LS-2 - Rotação para a esquerda de 2 posições
- ☐ K2 - Obtenção da segunda chave
- ☐ IP - Permutação Inicial
- ☐ E/P - Permutação de expansão
- ☐ XOR - OU exclusivo
- ☐ S0 & S1 - Substituições
- ☐ P4 - Permutação de 4 bits
- ☐ SW - Swap
- ☐ IP-1 - Permutação Inicial inversa

A primeira pergunta dessa seção destina-se à identificar quais etapas da execução do algoritmo possuem explicações que podem ser melhoradas, ou seja, não foram suficiente para compreensão completa da etapa pelo usuário.

Figura 44 – Melhorias nas execuções das etapas

Quais das etapas poderiam ter sua execução melhoradas? Marque todas aplicáveis.

- ☐ P10 - Permutação de 10 bits
- ☐ LS-1 - Circular Left Shift de 1 posição
- ☐ P8 - Permutação de 8 bits
- ☐ K1 - Obtenção da primeira chave
- ☐ LS-2 - Circular Left Shift de 2 posições
- ☐ K2 - Obtenção da segunda chave
- ☐ IP - Permutação Inicial
- ☐ E/P - Permutação de expansão
- ☐ XOR - OU exclusivo
- ☐ S0 & S1 - Substituições
- ☐ P4 - Permutação de 4 bits
- ☐ SW - Swap
- ☐ IP-1 - Permutação Inicial inversa

A segunda pergunta dessa seção destina-se à identificar quais etapas da execução do algoritmo possuem execuções (passo a passo) que podem ser melhoradas, ou seja, não foram suficiente para compreensão completa do passo a passo da etapa pelo usuário.

## 4.2 Respostas

TODO: Escrever...

### 4.2.1 Enquadramento do usuário

TODO: Escrever...

### 4.2.2 Conhecimento antes e depois da utilização do simulador

TODO: Escrever...

### 4.2.3 Efetividade do aprendizado

TODO: Escrever...

#### 4.2.4 Melhorias no simulador

TODO: Escrever...

## 5 Conclusões e Trabalhos futuros

TODO: Escrever...

### 5.1 Extensão do Simulador

- Adicionar outros algoritmos ao simulador.
- Adicionar o tema **escuro**.
- Disponibilizar o simulador em outras línguas além do **Português-BR** para aumentar a sua abrangência e efetividade.
- Melhorar a interface quando esta estiver sendo visualizada em *mobile*.



# Referências

- AUSUBEL, D. P.; NOVAK, J. D.; HANESIAN, H. Psicologia educacional. 2. ed. [S.l.]: Interamericana, 1980. ISBN 8520100848. 28, 29
- AVELINO, D.; AVELINO, I. C. Aplicações da criptografia em ambientes computacionais. In: . [S.l.]: IV SEGeT – Simpósio de Excelência em Gestão e Tecnologia, 2007. 17
- BANKS, J. *et al.* Discrete-Event System Simulation. 5. ed. [S.l.]: Prentice Hall, 2009. ISBN 9780136062127. 30
- BROCARD, M. L.; ROLT, C. R. D.; FERNANDES, R. Introdução à certificação digital: da criptografia ao carimbo de tempo. BRy Tecnologia, 2006. 18
- GAINES, H. F. Cryptanalysis: a study of ciphers and their solution. [S.l.]: Dover Publications, 1956. ISBN 9780486200972. 16
- HAMAWAKI, M. H.; PELEGRINI, C. de M. As ferramentas do ensino a distância e suas contribuições para a eficácia no processo de aprendizagem do aluno. CEPPG - n° 21, p. 84 à 91, 2009. ISSN 1517-847. 28, 29
- ITU, I. T. U. Security architecture for open systems interconnection for CCITT applications: Recommendation X.800. [S.l.]: The international telegraph and telephone consultative committee - CCITT, 1991. Geneva. 16, 17
- KNUDSEN, J. Java Cryptography. [S.l.]: O'Reilly, 1998. 16
- MAIA, L. P. SOsim: Simulador para o ensino de sistemas operacionais. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro - UFRJ, 2001. 12, 30
- MAIA, L. P.; PACHECO, A. C. A simulator supporting lectures on operating systems. 33'd ASEE/IEEE Frontiers in Education Conference, 2003. 12, 30
- SALOMON, D. Coding for Data and Computer Communications. [S.l.]: Springer, 2005. ISBN 9780387212456. 17
- STALLINGS, W. Cryptography and Network Security: Principles and Practices. [S.l.]: Prentice Hall, 2010. ISBN 0136097049. 26, 27, 28
- STALLINGS, W. Criptografia e Segurança de Redes: Princípios e Práticas. [S.l.]: Pearson, 2014. 16, 17, 19, 21, 23, 24, 25
- WIKIPÉDIA. Enigma (máquina). Wikipédia, 2020. Disponível em: <[https://pt.wikipedia.org/wiki/Enigma\\_\(máquina\)](https://pt.wikipedia.org/wiki/Enigma_(máquina))>. 19
- YOUNG, A.; YUNG, M. Malicious Cryptography: Exposing Cryptovirology. [S.l.]: Wiley Publishing, Inc., 2004. ISBN 9780764549755. 17